

## 四 网络层

网络层的目的是实现两个端系统之间的数据透明传送，具体功能包括寻址和路由选择、连接的建立、保持和终止等。它提供的服务使传输层不需要了解网络中的数据传输和交换技术

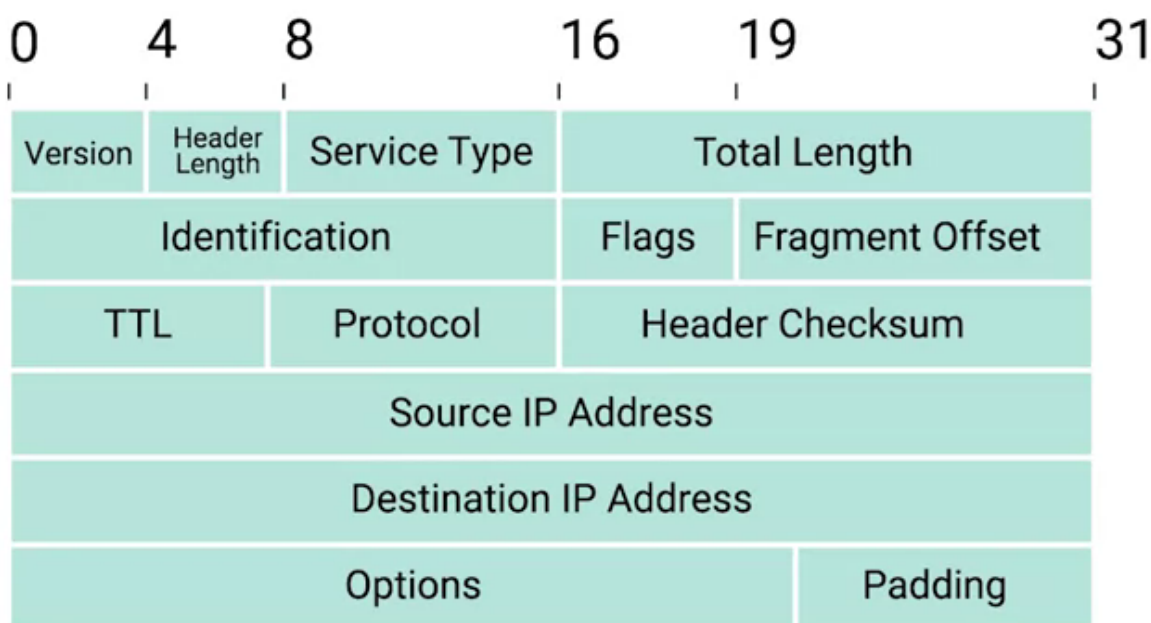
网络层主要是为传输层提供服务，为了向传输层提供服务，则网络层必须要使用数据链路层提供的服务。而数据链路层的主要作用是负责解决两个直接相邻节点之间的通信，但并不负责解决数据经过通信子网中多个转接节点时的通信问题，因此，为了实现两个端系统之间的数据透明传送，让源端的数据能够以最佳路径透明地通过通信子网中的多个转接节点到达目的端，使得传输层不必关心网络的拓扑构型以及所使用的通信介质和交换技术

### IP协议

IP协议是一种网络层协议，它负责将数据包从源设备传输到目标设备。IP协议是因特网上最基本和最重要的协议之一，它允许不同类型的计算机和其他设备通过一个统一的网络进行通信。IP协议的主要功能包括传输路由寻址和分包等。

### IPv4 数据报文结构

IPv4 数据报文是由一系列高度结构化的字段严格定义，IP 数据两个主要部分是 header 和 payload。



- **Version** - 第一个字段长度为 4 个二进制位，代表着 IP 协议的版本。常见的 IP 协议版本是 4，即 IPv4。
- **Header Length** - Header Length 字段长度为 4 个二进制位，代表着整个 header 的长度。如果是 IPv4，则 Header 的长度永远都是 20，事实上，20 个字节是 IP header 的最小长度，你不能在小于 20 自己的空间里合适的描述一个 IP Header。
- **Service Type** - Service Type 字段长度为 8 个二进制位，用来指定 QoS 技术的详细情况。QoS 的作用是允许路由器作出决策，在一系列 IP 数据报文中，选择出最为重要的一个数据报文。
- **Total Length** - Total Length 字段长度为 16 个二进制位，用来表示 IP 数据报文的整体长度。单个数据报文的最大长度为 16 个二进制位都为 1，即为 65,535。
- **Identification** - Identification 字段长度为 16 个二进制位，用来将消息分组在一起，当要发送的数据大于单个数据报文允许的最大值时，则 IP 层需要将原始的大的数据包分割成几个小的数据包，在这种情况下 Identification 字段用来被接收端标识分割后的 数据包属于同一个数据包。
- **Flag** - Flag 字段用来标识数据报文是否允许分段，或者标识数据报文已经分段。
- **Fragmentation** - 是将一个大的 IP 数据报文分割成多个小的数据报文的进程。
- **TTL** - TTL 字段的长度为 8 个二进制位，指定一个数据报文在经过多少个路由跳转后丢弃。
- **Protocol** - Protocol 字段的长度为 8 个二进制位，包含数据标识那个传输层的协议被使用，最常见的传输层协议是 TCP 或 UDP。
- **Header Checksum** - Header checksum 字段用来对整个 IP 数据报文 header 进行校验，它和 Ethernet Checksum 字段类似，通常由于 TTL 字段经过任意一个路由器时都会被修改，Header Checksum 字段相应的也会被修改。
- **Source IP Address** - 长度为 32 个二进制位，代表着源 IP 地址。
- **Destination IP Address** - 长度为 32 个二进制位，代表着目的地 IP 地址。
- **Option** - 可选的字段，用来设定一些特定字符，通常用于测试目的。
- **Padding** - 相当于一个占位符字段，由于 Option 字段时可选的一个变量，长度不定，该字段只是一些 0 串，用来确保 Header 的整体长度。

## IPv4 地址

IPv4 地址:

192.168.1.107 = 11000000.10101000.00000001.01101011

子网掩码:

255.255.255.0 = 11111111.11111111.11111111.00000000

Prefix: /24

11000000.10101000.00000001.01101011  
Network Host

IPv4 地址:

10.66.192.36 = 00001010.01000010.11000000.00100100

子网掩码:

255.255.0.0 = 11111111.11111111.00000000.00000000

Prefix: /16

00001010.01000010.11000000.00100100  
Network Host

- IPv4 地址长度为 32 为二进制数，由 4 组十进制数组成，4 组十进制数之间通过圆点连接
- IPv4 地址有两部分组成：网络部分(Network)和主机部分(Host)，同一子网的所有主机可以不经过路由而连通彼此，同一子网中主机部分唯一。
- 子网掩码用来区分网络部分和主机部分，如上图，10.66.192.36 子网掩码为 255.255.0.0，即前缀是 16，则为 10.66 网段。
- 广播地址：当主机部分所有为位置为1是就为广播地址，如上两个地址的广播地址分别为 192.168.1.255，10.66.255.255。
- IPv4 地址范围 0.0.0.0 - 255.255.255.255。

示例：IBM IP 地址，9 是网络地址，100.100.100 是主机地址

9.100.100.100

## 二进制和十进制转换

8 bit

$$2^8 = 256$$

0-255

4 bit

$$2^4 = 16$$

16 bit

$$2^{16} = 65536$$

## IPv4 地址分类

为了更好的管理互联网网络，IPv4 地址被分为五个类型：A、B、C、D、E，地址分类是从两个维度进行（或依赖两个原则）：

- 以第一个十进制数字的范围作为基准划分：
  - 0 - 127 为 A 类地址
  - 128 - 191 为 B 类地址
  - 192 - 223 为 C 类地址
  - 224 - 239 为 D 类地址
  - 240 - 255 为 E 类地址
- 以网络部分和主机部分作为基准的划分：
  - A 类地址只有第一组为网络地址，后面三组为主机地址
  - B 类地址前两组为网络地址，后两组为主机地址
  - C 类地址前三组为网络地址，最后一组为主机地址

对比 IPv4 地址的二进制表述和十进制表述可以帮助理解 IPv4 地址分类：


二进制表述					十进制表述				
	一位	二位	三位	四位		一位	二位	三位	四位
A 类地址	0				A 类地址	0 - 127			
B 类地址	10				B 类地址	128 - 191			
C 类地址	110				C 类地址	192 - 223			
D 类地址	1110				D 类地址	224 - 239			
E 类地址	1111				E 类地址	240 - 255			

类型	描述	范围	最大主机数	子网掩码
A	第一位十进制数用来做网络地址，后面三位十进制数用来做主机地址；以二进制表述，第一位以 0 开头，即二进制范围为 00000000 - 01111111	0.0.0.0 - 127.255.255.255	16 M	255.0.0.0/0xFF000000
B	前两位十进制数用来做网络地址，后面二位十进制数用来做主机地址；以二进制表述，第一位以 10 开头，即二进制范围为 10000000 - 10111111	128.0.0.0 - 191.255.255.255	64000	255.255.0.0/0xFFFF0000
C	前三位十进制数用来做网络地址，后面一位十进制数用来做主机地址；以二进制表述，第一位以 110 开头，即二进制范围为 11000000 - 11011111	192.0.0.0 - 223.255.255.255	254	255.255.255.0/0xFFFFF00

## 不能分配给网络设备地址

不是所有的 IP 地址可以分配给网络设备，如下一些地址属预留地址，不能分配给网络设备：

- **0.0.0.0**：代表所有网络
- **127.0.0.0 - 127.255.255.255**：loopback 本地测试地址
- **255.255.255.255**：代表所有主机
- 网络地址和广播地址
  - 一个网络中的第一个地址称为网络地址，用于标识一个网络，这个个地址是网络预留地址
  - 一个网络中最后一个地址称为广播地址，用于向该网络中所有主机发送数据的特殊地址
  - 例如 10.1.10.0/24 网络，10.1.10.0 是网络预留地址，10.1.10.255 是广播地址。

参照  了解更多关于网络分类。

## 不可路由的地址

不可路由的地址空间是一些 IP 范围，可以被任何人使用，但是不能路由。不是每台每台连接到 Internet 的计算机都需要能够与其他连接到 Internet 的计算机进行通信，不可路由的地址为这一需求而定，此类节点构成的网络他们可以相互通信，但没有网关路由器会尝试将流量转发到此类网络。

对应 IPv4地址范围，不可路由的地址空间主要有三个范围：

所属分类	网络	地址范围
A	10.0.0.0/8	10.0.0.0 - 10.255.255.255.255
B	172.16.0.0/12	172.16.0.0 - 172.31.255.255
C	192.168.0.0/16	192.168.0.0 - 192.168.255.255

## IPv4 子网划分

“有类编址”的地址划分过于死板，划分的颗粒度太大，会有大量的主机号不能被充分利用，从而造成了大量的IP地址资源浪费。因此可以利用子网划分来减少地址浪费，将一个大的有类网络，划分成若干个小的子网，使得IP地址的使用更为科学。那么我们来看一下如何完成子网划分。

如果一个 IPv4 地址 属于 A 类或 B类地址，则可能存在的最大主机较多，这就需要子网来进一步分组组成较小的网络，这就叫做子网。

## 子网掩码

子网掩码长度也为 32 位二进制数，通常由 4 组十进制数组成，4 组十进制数之间通过圆点连接，二进制表述，子网掩码由连续的 1 和 连续的 0 构成，通常子网掩码由十进制表述，例如下表为一些子网掩码二进制和十进制示例：

二进制	十进制
11111111.11111111.11111111.00000000	255.255.255.0
11111111.11111111.00000000.00000000	255.255.0.0
11111111.00000000.00000000.00000000	255.0.0.0
11111111.11111111.11111110.00000000	255.255.254.0
11111111.11111111.11111100.00000000	255.255.252.0
11111111.11111111.11111000.00000000	255.255.248.0
11111111.11111111.11110000.00000000	255.255.240

## CIDR(classless inter-domain routing)

- CIDR 是描述 IP 地址的一种更加灵活的方法，以斜杠 + 数字来表示掩码长度，这样对子网的划分更加易读。
- CIDR(classless inter-domain routing，无类别域间路由)采用IP地址加掩码长度来标识网络和子网，而不是按照传统A、B、C等类型对网络地址进行划分。
- CIDR容许任意长度的掩码长度，将IP地址看成连续的地址空间，可以使用任意长度的前缀分配，多个连续的前缀可以聚合成一个网络，该特性可以有效减少路由表条目数量。

<https://ipaddressguide.com/cidr>

二进制	十进制	CIDR
11111111.11111111.11111111.00000000	255.255.255.0	/24
11111111.11111111.00000000.00000000	255.255.0.0	/16
11111111.00000000.00000000.00000000	255.0.0.0	/8
11111111.11111111.11111110.00000000	255.255.254.0	/23
11111111.11111111.11111100.00000000	255.255.252.0	/22
11111111.11111111.11111000.00000000	255.255.248.0	/21

CIDR	Netmask	First IP	Last IP
10.1.10.0/30	255.255.255.252	10.1.10.0	10.1.10.3
10.1.10.4/30	255.255.255.252	10.1.10.4	10.1.10.7
10.1.10.8/30	255.255.255.252	10.1.10.8	10.1.10.11

10.1.10.12/30	255.255.255.252	10.1.10.12	10.1.10.15
10.1.10.16/30	255.255.255.252	10.1.10.16	10.1.10.19
10.1.10.20/30	255.255.255.252	10.1.10.20	10.1.10.23
10.1.10.24/30	255.255.255.252	10.1.10.24	10.1.10.27
10.1.10.28/30	255.255.255.252	10.1.10.28	10.1.10.31
10.1.10.32/30	255.255.255.252	10.1.10.32	10.1.10.35
10.1.10.36/30	255.255.255.252	10.1.10.36	10.1.10.39
10.1.10.40/30	255.255.255.252	10.1.10.40	10.1.10.43
10.1.10.44/30	255.255.255.252	10.1.10.44	10.1.10.47
10.1.10.48/30	255.255.255.252	10.1.10.48	10.1.10.51
10.1.10.52/30	255.255.255.252	10.1.10.52	10.1.10.55
10.1.10.56/30	255.255.255.252	10.1.10.56	10.1.10.59
10.1.10.60/30	255.255.255.252	10.1.10.60	10.1.10.63
10.1.10.64/30	255.255.255.252	10.1.10.64	10.1.10.67
10.1.10.68/30	255.255.255.252	10.1.10.68	10.1.10.71
10.1.10.72/30	255.255.255.252	10.1.10.72	10.1.10.75
10.1.10.76/30	255.255.255.252	10.1.10.76	10.1.10.79
10.1.10.80/30	255.255.255.252	10.1.10.80	10.1.10.83
10.1.10.84/30	255.255.255.252	10.1.10.84	10.1.10.87
10.1.10.128/30	255.255.255.252	10.1.10.128	10.1.10.131
10.1.10.240/30	255.255.255.252	10.1.10.240	10.1.10.243
10.1.10.244/30	255.255.255.252	10.1.10.244	10.1.10.247
10.1.10.248/30	255.255.255.252	10.1.10.248	10.1.10.251
10.1.10.252/30	255.255.255.252	10.1.10.252	10.1.10.255

CIDR	Netmask	First IP	Last IP
10.1.10.0/27	255.255.255.224	10.1.10.0	10.1.10.31
10.1.10.32/27	255.255.255.224	10.1.10.32	10.1.10.63
10.1.10.64/27	255.255.255.224	10.1.10.64	10.1.10.95
10.1.10.96/27	255.255.255.224	10.1.10.96	10.1.10.127
10.1.10.128/27	255.255.255.224	10.1.10.128	10.1.10.159
10.1.10.160/27	255.255.255.224	10.1.10.160	10.1.10.191



10.1.10.192/27	255.255.255.224	10.1.10.192	10.1.10.223
10.1.10.224/27	255.255.255.224	10.1.10.224	10.1.10.255

## 查看本机 IP 地址

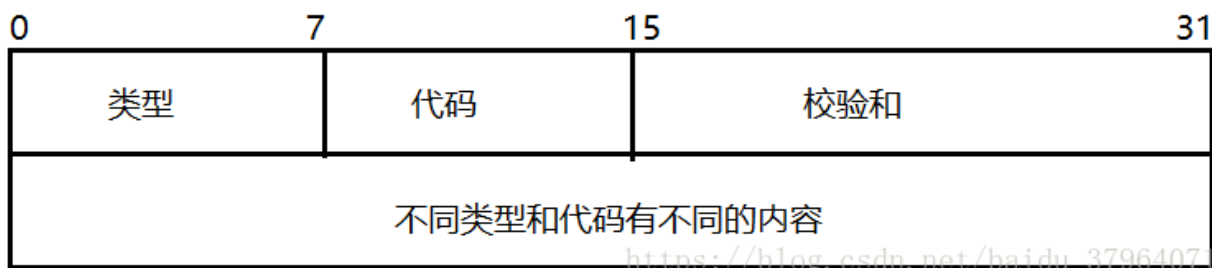
### MacOS and Linux

```
$ ifconfig |grep inet
    inet 127.0.0.1 netmask 0xff000000
    inet6 ::1 prefixlen 128
    inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
    inet6 fe80::aede:48ff:fe00:1122%en7 prefixlen 64 scopeid 0x4
    inet6 fe80::4a8:381f:ed4b:16fd%en0 prefixlen 64 secured scopeid 0x6
    inet 192.168.2.103 netmask 0xffffffff broadcast 192.168.2.255
    inet6 fe80::68bb:f8ff:fe22:4261%awdl0 prefixlen 64 scopeid 0x7
    inet6 fe80::68bb:f8ff:fe22:4261%llw0 prefixlen 64 scopeid 0x8
    inet6 fe80::79ef:6ac5:e1ac:8b4d%utun0 prefixlen 64 scopeid 0xe
    inet6 fe80::d8e:3eec:b8ef:ac7e%utun1 prefixlen 64 scopeid 0xf
    inet6 fe80::ce81:b1c:bd2c:69e%utun2 prefixlen 64 scopeid 0x10
    inet 10.201.65.114 --> 10.201.65.114 netmask 0xffffffff
    inet 192.168.56.1 netmask 0xfffff00 broadcast 192.168.56.255
```

## ICMP

ICMP(Internet Control Message Protocol)是IP协议的辅助协协议，用来在网络设备间传递各种差错和控制信息，对于收集各种网络信息、诊断和排除各种网络故障等方面起着至关重要的作用。

### ICMP packet Struct:



- 类型 - 占一字节，标识ICMP报文的类型。
- 代码 - 占一字节，标识对应ICMP报文的代码。它与类型字段一起共同标识了ICMP报文的详细类型。

- 校验和 - 这是对包括ICMP报文数据部分在内的整个ICMP数据报的校验和，以检验报文在传输过程中是否出现了差错。

## 常见消息类型

Type	Code	描述
0	0	Echo Reply
3	0	网络不可达
3	1	主机不可达
3	2	协议不可达
3	3	端口不可达
5	0	重定向
8	0	Echo Request

## 应用

ICMP 协议应用在许多网络管理命令中，下面以 ping 和 traceroute( Windows 系统下是 tracert ) 命令为例详细介绍 ICMP 协议的应用。

### 1. ping 命令使用 ICMP 回送请求和应答报文

在网络可达性测试中使用的分组网间探测命令 ping 能产生 ICMP 回送请求和应答报文。目的主机收到 ICMP 回送请求报文后立刻回送应答报文，若源主机能收到 ICMP 回送应答报文，则说明到达该主机的网络正常。

```
$ ping www.baidu.com
PING www.a.shifen.com (39.156.66.14) 56(84) bytes of data.
64 bytes from 39.156.66.14 (39.156.66.14): icmp_seq=1 ttl=42 time=6.60 ms
64 bytes from 39.156.66.14 (39.156.66.14): icmp_seq=2 ttl=42 time=6.24 ms
64 bytes from 39.156.66.14 (39.156.66.14): icmp_seq=3 ttl=42 time=6.25 ms
64 bytes from 39.156.66.14 (39.156.66.14): icmp_seq=4 ttl=42 time=6.24 ms
64 bytes from 39.156.66.14 (39.156.66.14): icmp_seq=5 ttl=42 time=6.27 ms
```

### 2. 路由分析诊断程序 traceroute 使用了 ICMP 时间超过报文

traceroute 命令主要用来显示数据包到达目的主机所经过的路径（路由器）。通过执行一个 traceroute 到对方主机的命令，返回数据包到达目的主机所经历的路径详细信息，并显示每个路径所消耗的时间。

服务器端查看 MAC 地址，验证与上面第 3 步中客户端 ARP 表中以及第 4 步抓包中获取的 MAC 地址是否相同

## Linux

```
$ ip addr show eth1 | grep ether
    link/ether 08:00:27:c3:0f:80 brd ff:ff:ff:ff:ff:ff
```

## MacOS

```
$ ifconfig | grep ether
    ether ac:de:48:00:11:22
    ether aa:66:5a:4d:d6:a8
    ether 88:66:5a:4d:d6:a8
```

## ARP

- ARP(Address Resolution Protocol) 协议用来通过特定的 IP 地址发现该 IP 地址对应的硬件设备的 MAC 地址。
- 通常网络设备都有一个 ARP 表，ARP 表中包含着一系列 IP 地址与 MAC 地址对应的条目。ARP 表中条目通常会在较短的时间后过期，以确保网络设备及时感知到网络的变更。

## 客户端查看 ARP 表

```
$ arp -e -i eth1
Address          HWtype  HWaddress      Flags Mask    Iface
192.168.33.1     ether   0a:00:27:00:00:05 C              eth1
server.example.com ether   08:00:27:c3:0f:80 C              eth1
```

## 路由

### 什么是路由

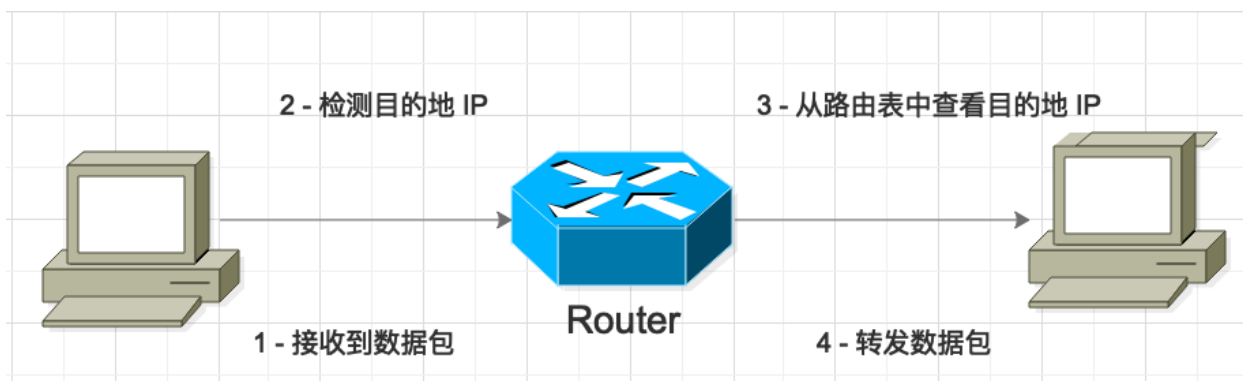
- 路由是指导报文转发的路径信息，通过路由可以确认转发IP报文的路径。
- 路由设备是依据路由转发报文到目的网段的网络设备，最常见的路由设备:路由器。
- 路由设备维护着一张路由表，保存着路由信息。

### 路由器

路由器是网络层设备（三层网络设备），它根据数据包的目的地址转发相应的数据包，将这一数据包的转发过程称为路由。一个路由器设备至少有两个网络接口，因为路由器工作机制至少需要连接两个网络。

## 路由的基本过程

下图描述了位于不同网络的 PC 通过路由器进行通信。数据包经过路由器转发到目的 PC 的过程就是路由的基本过程，具体包括四个步骤：



1. 路由器通过它的一个网络接口接收到一个数据包
2. 路由器检测数据包中目的地的 IP 地址（对源数据链路层以太网帧的头和尾去掉，只保留 IP 数据报文，从 IP 数据报文的头中获取目的地 IP 地址）
3. 路由器从路由表中查询目的地的 IP 地址
4. 路由器通过它的一个网络接口将数据包转发出去（修改 2 步骤中的 IP 数据报文，对 IP 数据报文头中的 TTL 字段减一，重新计算 Header Checksum 字段，然后封装一个新的以太网帧，添加头和尾）

**Note：** 如果数据包传输跨多个网络，则查询路由表或找出最近的一个网络将数据包转发出去，同时每经过一次路由，IP 数据报文的 TTL 字段都会减小 1。

## 路由表

路由表结构比较简单，通常有四个列：

1. Destination - 目的地网络，路由器上已知的所有网络都会存在一行，代表的是目的地的网络，包括网络地址和子网掩码。
2. Next Hop - 是去往目的网络最近的路由器的 IP 地址；如果去往目的网络不需要经过网络跳转，或者说目的地和路由器在同一个网络，则该字段是目的地的 IP 地址。

3. Total Hops - 这是了解路由以及路由表如何工作的关键部分，在任何复杂的网络（如 Internet）上，从一个点到另一个点都有很多不同的路径。

4. Interface - 路由器的网络接口，该接口用于将数据包从路由器转发出去

## Linux 上 route 命令查看路由信息

```
$ route -nv
Kernel IP routing table
Destination      Gateway          Genmask          Flags Metric Ref    Use Iface
0.0.0.0          10.1.10.2       0.0.0.0          UG      0      0      0 external
0.0.0.0          10.1.1.1        0.0.0.0          UG      9      0      0 mgmt
10.1.1.0         0.0.0.0         255.255.255.0    U      0      0      0 mgmt
10.1.10.0        0.0.0.0         255.255.255.0    U      0      0      0 external
10.1.20.0        0.0.0.0         255.255.255.0    U      0      0      0 internal
127.1.1.0        0.0.0.0         255.255.255.0    U      0      0      0 tmm
127.7.0.0        127.1.1.253     255.255.0.0      UG      0      0      0 tmm
127.20.0.0       0.0.0.0         255.255.0.0      U      0      0      0 tmm_bp
```

## Linux 上 ip route 查看路由信息

```
$ ip route list
default via 10.1.10.2 dev external
default via 10.1.1.1 dev mgmt metric 9 mtu 1500
10.1.1.0/24 dev mgmt proto kernel scope link src 10.1.1.245
10.1.10.0/24 dev external proto kernel scope link src 10.1.10.240
10.1.20.0/24 dev internal proto kernel scope link src 10.1.20.240
127.1.1.0/24 dev tmm proto kernel scope link src 127.1.1.254
127.7.0.0/16 via 127.1.1.253 dev tmm
127.20.0.0/16 dev tmm_bp proto kernel scope link src 127.20.0.254
```

路由表中路由条目获取有三种方式：

- **直连路由** - 由设备自动生成指向本地直连网络
- **静态路由** - 由网络管理员手工配置的路由条目
- **动态路由** - 路由器运行动态路由协议学习到的路由

## 路由协议

路由协议主要目的有两个：

1. 网络发现
2. 路由表更新

路由协议可以分为两类：

1. IGP(Interior Gateway Protocol 内部网关协议) - IGP 通常是在一个自治系统（Autonomous system, AS，一个，有时是多个实体管辖下的所有 IP 网络和路由器的全体，例如一个企业/组织

的内网) 内路由器共享信息

2. EGP(Exterior Gateway Protocol 外部网关协议) - EGP 是自制系统之间路由器共享信息。

IGP 协议可以进一步分为两类：

1. 链路状态路由协议 (Link State Routing Protocol)
2. 距离矢量路由协议 (Distance-Vector Protocol)

## 动态路由协议

现代路由器设备通常通过动态路由器共享远程网络的状态和可达性，如下是一些常见的动态路由协议

- RIP(Routing Information Protocol)
- EIGRP(Enhanced Interior Gateway Routing Protocol)
- OSPF(Open Shortest Path First)
- BGP(Border Gateway Protocol)

动态路由协议的优点是：

- 动态更新路由表
- 不仅仅针对不同的网络可以选择出一个最佳路径，而且在初始最佳路径不可用（网络拓扑变化）后可以重新选择一个最佳路径
- 不同路由器之间动态共享路由信息，而不需要网络管理员人为参与

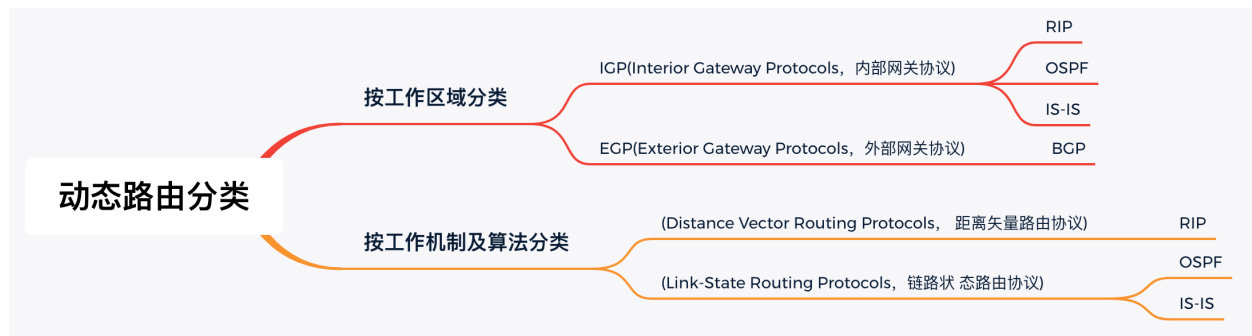
## OSPF

### 概述

由于静态路由无法适应规模较大的网络且无法动态响应网络变化，就有了动态路由协议。静态路由协议有以下问题：

1. 无法适应规模较大的网络
2. 无法动态响应网络变化

动态路由分类：

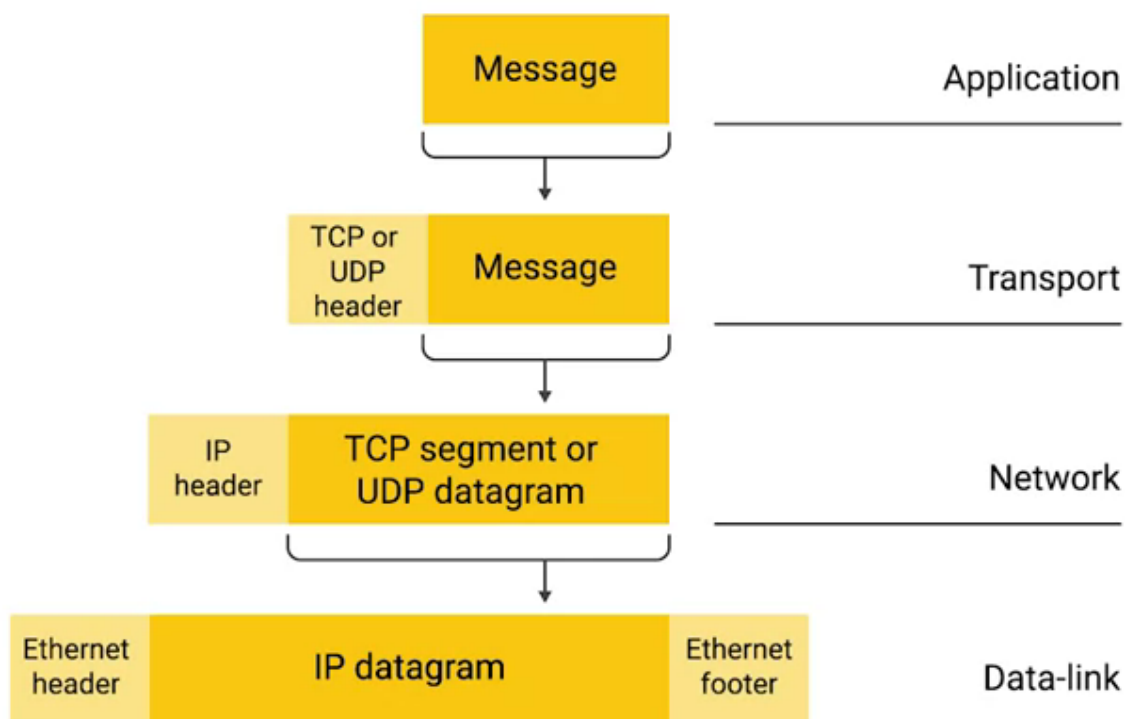


OSPF是典型的链路状态路由协议，是目前业内使用非常广泛的IGP协议之一：

- OSPF是典型的链路状态路由协议，是目前业内使用非常广泛的IGP协议之一。
- 目前针对IPv4协议使用的是OSPF Version 2(RFC2328);针对IPv6协议使用OSPF Version 3(RFC2740)。如无特殊说明本章后续所指的OSPF均为OSPF Version 2。
- 运行OSPF路由器之间交互的是LS(Link State，链路状态)信息，而不是直接交互路由。LS信息是OSPF能够正常进行拓扑及路由计算的关键信息。
- OSPF路由器将网络中的LS信息收集起来，存储在LSDB中。路由器都清楚区域内的网络拓扑结构，这有助于路由器计算无环路径。
- 每台OSPF路由器都采用SPF算法计算达到目的地的最短路径。路由器依据这些路径形成路由加载到路由表中。
- OSPF支持VLSM(Variable Length Subnet Mask，可变长子网掩码)，支持手工路由汇总。
- 多区域的设计使得OSPF能够支持更大规模的网络。

## Fragmentation

如下图，应用层发送一个消息在网络模型中每一层封装过程，底层包的 payload 是临近上一层包，



1. 数据链路层 Ethernet 帧的 Payload 是其上一层网络层 IP 数据报文
2. 网络层 IP 数据报文的 Payload 是其上一层传输层 TCP 报文或 UDP 报文
3. 传输层 TCP/UDP 报文的 Payload 是其上一层应用层的 Message

如果 IP Datagram 的大小大于当前网络允许的 MTU(Maximum Transmission Unit) 时，则 IP Datagram 被首先分割成多个 Packet，然后在网络上传输，这个过程叫做 **Fragmentation**。

**Fragmentation** 可以发生在初始的主机，或在路由过程中。

**Fragmentation** 可能会造成一个重传的出现，例如如果一个 Packet 的丢失，可能会导致多个 IP Datagrams 的重传。

| Note: 以太网上允许的最大 MTU 默认值为 1500 bytes。